Approximate Distributed Kalman Filtering in Sensor Networks with Quantifiable Performance

Demetri P. Spanos
Control and Dynamical Systems
California Institute of Technology
Email: demetri@calttech.edu

Reza Olfati-Saber
Mechanical and Aerospace Eng.
University of California, Los Angeles
Email: olfati@seas.ucla.edu

Richard M. Murray
Control and Dynamical Systems
California Institute of Technology
Email: murray@caltech.edu

***Abstract*— We analyze the performance of an approximate distributed Kalman filter proposed in recent work on distributed coordination. This approach to distributed estimation is novel in that it admits a systematic analysis of its performance as various network quantities such as connection density, topology, and bandwidth are varied. Our main contribution is a frequency-domain characterization of the distributed estimator's steady-state performance; this is quantified in terms of a special matrix associated with the connection topology called the *graph Laplacian*, and also the rate of message exchange between immediate neighbors in the communication network.**

I. Introduction

The possibility of large decentralized sensor networks has renewed interest in parallel and distributed signal processing, especially as regards tracking and estimation. Kalman filters form the bulk of these applications, and admit various levels of decentralization under appropriate assumptions. However, classical work on distributed Kalman filters typically assumes perfect instantaneous communication between every node on the network and every other node. While the resulting algorithms remain immensely useful even for practical networks, they do not allow any straightforward analysis of the degradation of their performance when communication is limited.

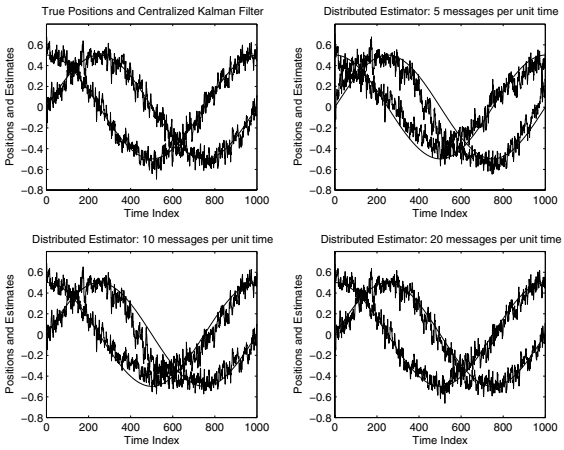

Fig. 1. Typical behavior of the distributed Kalman filter as the number of message exchanges increases. This paper provides analysis that allows one to quantify this behavior as a function of network topology, bandwidth (or messaging rate), and connection density.

Recent work in the control and systems community has examined a strategy for dynamic iterative Kalman filtering. This approach implements a distributed filter in which each node dynamically tracks the instantaneous least-squares fusion of the current input measurements. This allows the nodes to run independent local Kalman filters using the globally fused input, and (asymptotically) obtain the performance of a centralized Kalman filter. The fact that only *inputs* (not *estimates*) are shared allows a frequency-domain analysis of the performance of this distributed estimation scheme. This characterization allows one to understand the behavior depicted in Figure 1 in a quantitative way.

The main contribution of this article is a transfer function describing the error behavior of the distributed Kalman filter in the case of stationary noise processes. The magnitude of this transfer function goes to zero exponentially as the speed of the communication network relative to the speed of the estimated process becomes large. Specifically, we will show that the following quantity is particularly relevant:

$$\left(1 - \frac{\lambda_2}{d_{\max} + 1}\right)^n .$$

Here, $d_{\max}$ is the maximal node-degree, λ_2 is the *algebraic connectivity* of the network (defined in the following section), and n is the number of neighbor-to-neighbor message exchanges allowed per update of the estimation process.

II. Motivation

Kalman filtering is a fundamental tool in tracking and estimation, and it will be essential to provide this functionality in sensor networks. Multi-sensor filtering is fundamentally about information propagation, and sensor networks pose important challenges in the speed, reliability, and cost of this propagation. Power limitations will force designers to keep communication to a minimum, and perhaps also to put nodes into "sleep" modes intermittently. Simultaneously, one will want maximum estimation performance subject to the constraints imposed by the network technology. Since the Kalman filter is an intrinsically real-time algorithm, one would like to understand the impact of the network parameters on the real-time performance of distributed approximations to the optimal filter. This article provides such an analysis for a recent design proposed in [1]; the intuitive explanation for this analysis is based on the interconnection structure shown in Figure 2, and amounts to understanding the transfer function of a distributed low-pass filter.

Distributed and decentralized estimation has attracted much attention in the past, and there is a large associated literature. The classic work of Rao and Durrant-Whyte [2] presents an approach to decentralized Kalman filtering which accomplishes globally optimal performance in the case where all sensors can communicate with all other sensors. Further, this design "fails gracefully" as individual sensors are removed from the network due to its distributed design. However, it is difficult to understand the performance of this algorithm when point-to-point communication between each pair of nodes is unavailable, as is likely to be the case in a large-scale sensor network.

Much recent research effort has been dedicated to understanding the networking and computational challenges associated with large

0-7803-9201-9/05/$20.00 ©2005 IEEE

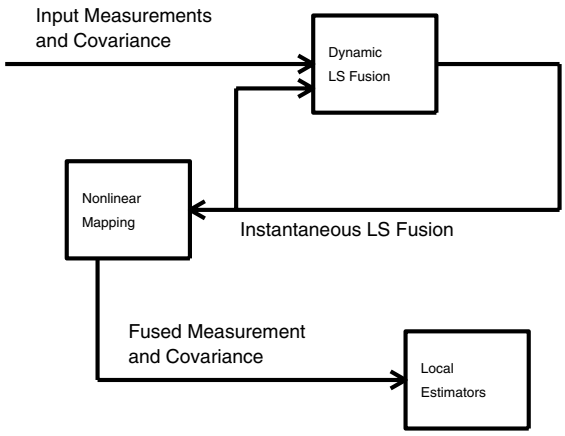

Fig. 2. The structure of the distributed Kalman filter proposed in Spanos, Olfati-Saber, and Murray [1].

sensor networks having only limited communication and routing capabilities. The work of Estrin, Govindan, Heidemann, and Kumar in [3], as well as that of Akyildiz, Su, Sankarasubramniam, and Cayirci [4] present excellent surveys of the challenges associated with this new technology. The work of Zhao, Shin, and Reich [5] addresses similar challenges in dynamically fusing the information collected by a large network of sensors, while incorporating the costs associated with excessive communication and computation. This problem has significant implications for networking protocols; this aspect of sensor networks is addressed in the work of Heinzelman, Kulik, and Balakrishnan [6].

The dynamics of coordination mechanisms in networks has attracted much attention in the control and systems community; we refer the reader to the works of Olfati-Saber and Murray [7], Jadbabaie, Lin, and Morse [8], and references therein for an introduction to recent developments in this area. The former article presents a decentralized "diffusion" mechanism for obtaining weighted averages of individual agent inputs in the face of delays and link loss. The work of Mehyar *et al.* [9] shows that this can be successfully translated to a truly asynchronous peer-to-peer system operating on a TCP/IP network[1]. Finally, the averaging mechanism is generalized to develop real-time tracking of optimally fused least-squares estimates and an associated decentralized Kalman filter in Spanos, Olfati-Saber, and Murray [1].

The convergence performance of these diffusion-based designs depends on the *algebraic connectivity* of the network, which is the smallest positive eigenvalue of the associated *Laplacian* matrix (see the article by Merris [10] for graph-theoretic fundamentals regarding the Laplacian). In the case where centralized topology information is available *a priori*, the work of Xiao and Boyd [11] provides very useful results for optimizing this convergence rate. Additional work by Xiao, Boyd, and Lall [12] provides an analogous rate optimization for the static least-squares fusion problem.

III. Notation and Assumptions

We consider a set V of N sensor nodes, each labeled $i = 1, 2, \ldots, N$. These sensors communicate on a network modeled as a graph $G = (V, E)$, where the edge (i, j) is in E if and only if nodes i and j can exchange messages with each other. We denote the neighborhood of node i by N_i. Note that this can represent either the physical communication links or some other overlay network imposed with routing. We again remark that this mechanism can be generalized to a realistic fully asynchronous version (see [9]).

We assume that the graph G is connected, and for the sake of this paper, static. See the work in [7], [13], [9], and [14] for extensions of this mechanism to switching-topologies, randomly failing links, asynchronous peer-to-peer operation, and arbitrary splitting and merging of subnetworks.

A global physical process with state $\mathbf{p} \in \mathbf{R}^m$ evolves according to the discrete-time system

$$\mathbf{p}(t+1) = A\mathbf{p} + \mathbf{w}(t)$$

where $A \in \mathbf{R}^{m \times m}$ and $\mathbf{w}(t)$ is zero-mean Gaussian noise with covariance matrix Q_w. The process initial condition is also distributed as a multivariate Gaussian with expectation $\mathbf{p}_0$ and covariance Q_0. We suppose that the process parameters A, Q_w, Q_0 and $\mathbf{p}_0$ are all available to every member of the network, in order to run local Kalman filters.

Each sensor takes measurements of the physical process according to the equation

$$\mathbf{y}_i(t) = \mathbf{p}(t) + \mathbf{n}_i(t).$$

The noise processes $\mathbf{n}_i(t) \in \mathbf{R}^{m \times m}$ are each independent zero-mean Gaussian with covariance $Q_i(t)$.

Our central assumption is that the network is "at least as fast as the physical process", in the sense that for each physical update index t, the network carries out $n > 1$ message exchanges on each edge. The work in [1] presents a mechanism for message exchange and decentralized estimation that is equivalent to a purely local Kalman filter for $n = 0$, and achieves the performance of the global Kalman filter in the limit as n becomes large. This result stems from the independence of the noise processes, which implies that it is sufficient to perform the spatial fusion before the time-propagation. It is thus sufficient for each sensor to run a local Kalman filter, taking as inputs the instantaneous spatial least-squares fusion of the input measurements

$$\mathbf{y}_{LS}(t) = \left(\sum_{i \in V} Q_i^{-1}(t) \right)^{-1} \sum_{i \in V} Q_i^{-1}(t)\mathbf{y}_i(t)$$

and the associated spatially-fused covariance

$$Q_{LS}(t) = \left(\sum_{i \in V} Q_i^{-1}(t) \right)^{-1}.$$

The distributed filter proposed in [1] provides a mechanism for tracking the average of the inverse-covariance-weighted measurements

$$\bar{\mathbf{y}}(t) = \frac{1}{N} \sum_{i \in V} Q_i^{-1}(t)\mathbf{y}_i(t)$$

and the time-varying average inverse-covariance

$$\bar{Q}(t) = \frac{1}{N} \sum_{i \in V} Q_i^{-1}(t) \quad .$$

Clearly, these two quantities are sufficient to reconstruct $\mathbf{y}_{LS}(t)$ by solving a linear system of equations at each time t. Further, knowledge of the number of nodes (available, for example, from a

[1] An extension to a fully asynchronous lossy broadcast channel is available, but it is fairly technical. Thus, despite its obvious relevance to sensor networks, we omit discussion of this version of the algorithm due to length limitations.

distributed minimum spanning tree) allows one to find the associated covariance signal.

The algorithm requires each node to maintain a vector variable $\mathbf{x}_i(t) \in \mathbf{R}^m$ and a matrix variable $M_i(t) \in \mathbf{R}^{m\times m}$. These are all initialized to $Q_i^{-1}(0)\mathbf{y}_i(0)$ and $Q_i^{-1}(0)$ respectively. The algorithm run at each node is as follows:

for each time t

$\mathbf{x}_i \leftarrow \mathbf{x}_i + Q_i^{-1}(t)\mathbf{y}_i(t) - Q_i^{-1}(t-1)\mathbf{y}_i(t-1)$

$M_i \leftarrow M_i + Q_i^{-1}(t) - Q_i^{-1}(t-1)$

for $k = 1, 2, \ldots, n$

$\mathbf{x}_i \leftarrow \mathbf{x}_i + \gamma \sum_{j\in N_i}(\mathbf{x}_j - \mathbf{x}_i)$

$M_i \leftarrow M_i + \gamma \sum_{j\in N_i}(M_j - M_i)$

end

end

It was shown in [1] that this algorithm makes each $\mathbf{x}_i$ and M_i track $\bar{\mathbf{y}}$ and $\bar{Q}$ respectively and so each node can thus locally compute $M_i^{-1}\mathbf{x}_i$ and $(NM_i)^{-1}$, treating these as approximations to $\mathbf{y}_{LS}$ and Q_{LS}. The algorithm described tracks $\bar{\mathbf{y}}$ and $\bar{Q}$ with zero error in "steady-state"[2]. This asymptotic result holds for arbitrary network interconnection and for arbitrary n, but the transient performance of the system depends on the network topology, connection density, and the number of messages per unit time n (a proxy for bandwidth).

The parameter γ is a stepsize, and must be chosen to ensure stability of the updating scheme. This is somewhat tricky in that stability can, in principle, depend on the graph structure of the network. A necessary and sufficient condition for stability under arbitrary interconnection of the sensors is $\gamma d_{\max} < 1$, where $d_{\max}$ is the maximum node-degree in the network. There is a "natural" choice

$$\gamma = \frac{1}{d_{\max}+1}$$

which has the property that if every sensor is connected to every other sensor, the global Kalman filter performance is recovered with a single message exchange per unit time, i.e. even with $n = 1$. Thus, with γ as above and complete interconnection this scheme is equivalent to that of Rao and Durrant-Whyte [2]. We will assume hereafter that γ is chosen in this way. This will only affect the constants entering the expressions to come, and not any of the qualitative results.

Finally, we will make use of the *Laplacian* matrix associated with the graph G. The Laplacian is defined as follows:

$$\begin{aligned} L_{ij} &= -1 \text{ if } (i,j) \in E, \text{ else } 0 \\ L_{ii} &= -\sum_{j\neq i} L_{ij}. \end{aligned}$$

This is a symmetric positive-semi-definite matrix, and the assumption that G is connected implies that L has exactly one zero eigenvalue and associated eigenvector $\mathbf{1}$ (the vector of all ones). Thus, repeated multiplication of a vector by $(I-\gamma L)$, where I is the $N\times N$ identity, drives each component of the vector to the average of the components of the initial vector (see [7]).

Note that each component of the $\mathbf{x}_i$ and M_i variables is updated independently. If for any one such component, we consider all the associated values across the network stacked in a vector $\mathbf{v} \in \mathbf{R}^N$, then the action of the inner update loop can be viewed as the following multiplication:

$$\mathbf{v} \leftarrow (I - \gamma L)^n \mathbf{v}. \tag{1}$$

The inner loop is precisely a Laplacian updating scheme for tracking the instantaneous average of the covariance-weighted measurements and the inverse-covariance matrices. Thus, the eigenvalues of the Laplacian matrix determine the convergence properties of the inner update loop. In particular, the smallest positive eigenvalue of the Laplacian, denoted λ_2, allows us to derive a bound on the worst-case convergence rate. This quantity is known in graph theory as the *algebraic connectivity*, and is strongly tied to connectivity properties of the graph (see [10] for a comprehensive exposition).

IV. Performance Analysis

In this section we will show a transfer function characterizing the performance of the distributed estimator in the case where the noise covariance has reached steady-state, i.e. all the covariance matrices $Q_i(t)$ are hereafter assumed constant, and we assume that the update loop for the M_i matrices has converged. This may seem a trivializing assumption in the context of sensor networks where estimated processes are likely to exhibit non-stationary statistics, and so some comments are in order.

First, let us provide some intuition for the distributed estimation scheme. At each time instant, each node has an estimate of the globally fused measurement inputs, and the globally fused covariance. This allows the sensor to implement an *approximation* to the global Kalman filter. For stationary noise, the global Kalman filter is just a Linear Time-Invariant (LTI) system parametrized by the covariance matrix and the process parameters. If the covariance matrices reach a limit, the matrices M_i converge exponentially (in time) to the average inverse covariance, and so each node rapidly "discovers" the covariance matrix associated with the global steady-state Kalman filter.

The distributed filter is inherently adaptive; if the covariance matrices begin changing again, approaching another steady-state value, the algorithm automatically tracks this change and finds the new covariance matrix to be used in the Kalman filter. Thus, analysis of the steady-state case is justified, either for slowly-varying error statistics, or for processes where the time-variation of the statistics is "bursty", remaining constant for large periods of time (relative to the update time-scale of the network).

Now, let us denote the transfer function of the global steady-state Kalman filter $K(z)$, an $m \times m$ matrix of transfer functions (determined by Q_{LS}); under the assumptions of this section, each sensor has already calculated Q_{LS} and can thus implement this filter locally. The nominal input to the filter is $\mathbf{y}_{LS}(t)$, but each node must instead use the following local estimate as input:

$$M_i^{-1}\mathbf{x}_i(t) = NQ_{LS}\mathbf{x}_i(t).$$

Since the quantity NQ_{LS} is just a constant matrix-gain for the steady-state filter, it suffices to examine the dynamics of the local estimates $\mathbf{x}_i(t)$, and how these variables relate to the "desired value" $(NQ_{LS})^{-1}\mathbf{y}_{LS}$. In order to do so, let us introduce the notation $\tilde{\mathbf{y}}$ and $\tilde{\mathbf{x}}$ denoting the stacked vectors of the $\mathbf{y}_i(t)$ and $M_i^{-1}\mathbf{x}_i(t)$ vectors, i.e.

$$\begin{aligned} \tilde{\mathbf{y}} &= \left[\mathbf{y}_1^T, \mathbf{y}_2^T \ldots \mathbf{y}_N^T\right]^T \\ \tilde{\mathbf{x}} &= \left[M_1^{-1}\mathbf{x}_1^T, M_2^{-1}\mathbf{x}_2^T \ldots M_N^{-1}\mathbf{x}_N^T\right]^T. \end{aligned}$$

[2]Here "steady-state" means that both the measurements and the covariance matrices approach a limit as $t \to \infty$. For example, this assumption is reasonable when estimating moving objects that occasionally halt for significant periods of time.

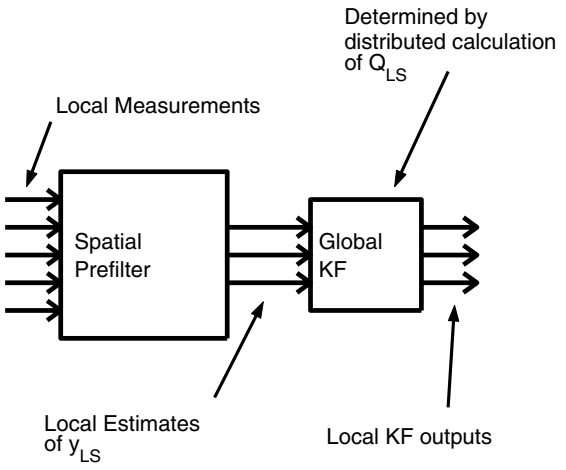

Fig. 3. The steady-state behavior of the distributed filter is equivalent to a global Kalman filter with a prefiltered input. The performance analysis of this article is based on quantifying how close the prefilter is to unity.

Here the superscript T denotes transposition. Note that when the covariance matrices are constant in time, the nominal input $\mathbf{y}_{LS}$ is related to the vector $\tilde{\mathbf{y}}$ by a constant matrix multiplication:

$$\mathbf{y}_{LS} = P_{LS}\tilde{\mathbf{y}} = Q_{LS} \begin{bmatrix} Q_1^{-1} & Q_2^{-1} & \dots & Q_N^{-1} \end{bmatrix} \tilde{\mathbf{y}}.$$

Now, the local estimates $M_i^{-1}\mathbf{x}_i$ are just the outputs of the spatial averaging filter described in the previous section. Specifically, the inputs to this filter are the local covariance matrices and the local measurements; in general the spatial averaging filter is nonlinear in an input-output sense because of the input nonlinearity $Q_i^{-1}(t)\mathbf{y}_i(t)$ and the output nonlinearity $M_i^{-1}\mathbf{x}_i(t)$.

However, when the covariance matrices are constant and the update of the M_i matrices has converged, the overall input-output behavior of the spatial averaging filter is *linear* as a mapping from the local measurement signals $\mathbf{y}_i(t)$ to the local estimate signals $M_i^{-1}\mathbf{x}_i(t)$. Thus, there is some $Nm \times Nm$ matrix of transfer functions, call it $H(z)$, such that

$$\tilde{\mathbf{x}}(z) = H(z)\tilde{\mathbf{y}}(z).$$

This lets us make a simple but intuitively useful statement:

> *In steady-state, the performance loss associated with the distributed estimation design is equivalent to premultiplication of the global Kalman filter by a low-pass filter determined by the network topology and speed.*

This situation is depicted in Figure 3. In order to quantify the performance loss, we simply need to understand the frequency response of this low-pass filter. We will do so by characterizing the *error transfer function*, which is a high-pass filter.

To do so, let us recall that each $M_i^{-1}\mathbf{x}_i$ tracks $\mathbf{y}_{LS}$ with zero steady-state error. This implies that the DC gain of H is just

$$H(1) = \begin{bmatrix} P_{LS}^T & P_{LS}^T & \dots & P_{LS}^T \end{bmatrix}^T.$$

Now, we want to consider the the error signal

$$\mathbf{e} = \tilde{\mathbf{x}} - \begin{bmatrix} \mathbf{y}_{LS}^T & \mathbf{y}_{LS}^T & \dots & \mathbf{y}_{LS}^T \end{bmatrix}^T,$$

and the transfer function from $\tilde{\mathbf{y}}$ to $\mathbf{e}(z)$

$$H_{\mathbf{e}\tilde{\mathbf{y}}}(z) = H(z) - \begin{bmatrix} P_{LS}^T & P_{LS}^T & \dots & P_{LS}^T \end{bmatrix}^T.$$

Note that this transfer function is zero at $z = 1$ by construction. Furthermore, we know that this system has the structure of m decoupled subsystems (one for each component of $\mathbf{x}_i$). Up to a constant matrix scaling determined by the covariance matrices, each such subsystem has transfer function of the following form

$$G(z) = \frac{1}{N}\mathbf{1}\mathbf{1}^T - (1-z^{-1})(I-\gamma L)^n \left[I - z^{-1}(I-\gamma L)^n \right]^{-1}.$$

This follows from the inner-loop Laplacian update operation (1), and the first-order differencing operation in the outer loop. We will further decompose these subsystems by exploiting the fact that the Laplacian is a symmetric matrix, and admits a spectral decomposition

$$L = 0 \cdot \mathbf{1}\mathbf{1}^T + \sum_{i>1} \lambda_i P_i$$

where the P_i terms are orthogonal projections onto mutually orthogonal subspaces and the λ_i terms are strictly positive eigenvalues (ordered from smallest to largest). Recall that the first term, corresponding to the nullspace of L, is known *a priori* because of the structure of the Laplacian matrix. Applying this formula for L in the above equation, we obtain

$$G(z) = \sum_{i>1} \frac{(1-\gamma\lambda_i)^n(z-1)}{z-(1-\gamma\lambda_i)^n} P_i.$$

Note that all of these terms are zero at $z = 1$, in accordance with our previous statement regarding $H_{\mathbf{e}\tilde{\mathbf{y}}}(z)$.

We have now decomposed the error transfer function (up to a block-diagonal matrix scaling) into Nm independent subsystems, each with trivial pole-zero structure. Specifically, they all share a common zero at $z = 1$, and each have a single pole of the form $z = (1-\gamma\lambda_i)^n$. Our assumption regarding γ implies that the largest such term is $(1-\gamma\lambda_2)^n$. This allows us to bound the error transfer function as follows:

$$\|H_{\mathbf{e}\tilde{\mathbf{y}}}(z)\| \leq \frac{C(1-\gamma\lambda_2)^n(z-1)}{z-(1-\gamma\lambda_2)^n} \tag{2}$$

where C is a constant determined by the covariance matrices.

V. The Impact of the Network: Topology, Density, and Bandwidth

The bound we have derived in the previous section allows us to quantify the performance of the distributed estimation scheme as a function of the network parameters λ_2, γ, and n. As a simple verification of our claim that the distributed scheme reduces to perfect estimation under complete interconnection, we will make use of the fact that for a complete graph

$$\lambda_i = d_{\max} + 1$$

for all $i > 1$. Combining this fact, our choice of γ from before, and the bound from the previous section, we obtain

$$H_{\mathbf{e}\tilde{\mathbf{y}}} = 0$$

for all $n > 1$, which implies that the global Kalman filter performance is achieved with a single message exchange on each link per unit time.

In general, we can understand the performance of this system by the following quantity:

$$\left(1 - \frac{\lambda_2}{1+d_{\max}}\right)^n.$$

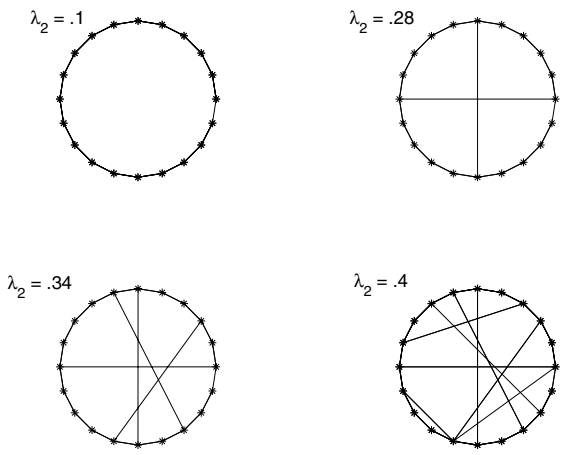

Fig. 4. The algebraic connectivity λ_2 for a few graphs. This quantity plays a central role in analyzing the performance of the distributed estimator.

As this quantity tends to zero, the performance of the distributed estimator approaches that of a centralized Kalman filter. Specifically we can study this quantity as a function of the three factors that are likely to vary across real-world sensor networks: topology, connection density, and bandwidth. The first aspect is captured in the eigenvalues of the Laplacian matrix, and in particular the *algebraic connectivity* λ_2. A comprehensive explanation of this quantity is beyond the scope of this paper (we again refer the reader to Merris [10]), but we can build some intuition with a simple example: a ring topology to which we sequentially add long-distance links. As more long-distance links are added, the algebraic connectivity grows, indicating better performance for the distributed Kalman filter (see Figure 4). This suggests an interesting use for routing in sensor networks, relative to the distributed estimation scheme: routing can be used to implement a few long-distance connections in order to improve λ_2 (see [15] for much more on this subject). In addition, for topologies that are fixed and known *a priori*, we also remind the reader of the results in [11] and [12] which allow one to optimize λ_2 using semi-definite programming.

The second aspect one must consider in the network is the density of connections. This has a dual effect on the distributed estimator. First, high connection-density influences the eigenvalues of the Laplacian matrix in relatively complicated ways, but overall it tends to influence the *large* eigenvalues more than the small ones. Second, it limits the stepsize parameter γ due to stability concerns. Thus, if one has control over the topology on which this distributed estimation scheme will be implemented, care should be taken to balance "high-connectivity" in the sense of λ_2 against small stepsize, as parametrized by the reciprocal of the maximum degree.

Finally, we see the dominating influence of the connection bandwidth, as represented by n. As n increases, the magnitude of the error transfer function shrinks exponentially. Considered in the light of a low-pass prefilter multiplying the Kalman filter, as n becomes large, the prefilter rapidly approaches unity for all frequencies.

VI. Simulation Example: A Sonar Array

This section presents simulations for the distributed filter on an array of sonar-like sensors in a *non-stationary* noise environment, in order to illustrate that the qualitative behavior indicated by the analysis of the previous section carries over to the non-stationary case. Note that we do not make any claim about analytical performance bounds for the non-stationary case, but the simulations indicate seemingly good correlation between the estimator's non-stationary performance and the stationary error bounds.

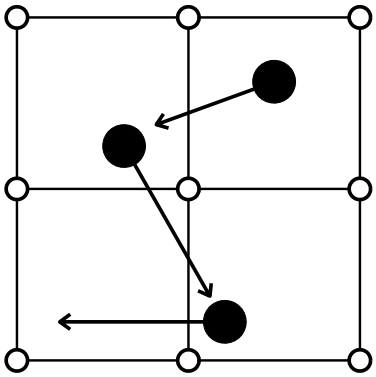

Fig. 5. A solid object moving through an array of sonar-like sensors. The lines indicate communication links between the individual sensors. The variance of the measurements taken by each sensor increases with distance.

The sensors report range and bearing at each time instant, with the variance of the bearing measurement set at ten times the variance in the range measurement. The range variance increases quadratically with distance from the target. The target moves in a circle centered at the central sensor, and the process model is a discretized double-integrator driven by white Gaussian noise. The measurements taken by the sensors are deliberately made very noisy (see Figure **??**) to illustrate the performance of this algorithm in an adversarial setting.

The simulations are carried out using the nine-node sensor network depicted in Figure 5, with γ chosen as $\frac{1}{1+d_{\max}}$. We show the results for two topologies, one as shown in Figure 5, and one where we add all the "diagonal" connections (i.e. very limited local routing). We simulate the algorithm for $n = 5, 10, 20$ message exchanges per estimator update[3] and show these results alongside the results from a centralized implementation of the Kalman filter. These are shown in Figures 6 and 7 (the trajectories shown in each figure are chosen from the sensor with the worst mean-square error). Figures 8 and 9 show the associated error transfer functions, based on the analysis in Section IV.

VII. Summary and Conclusions

We have examined the performance of an approximate distributed Kalman filter based on an iterative spatial averaging algorithm. This algorithm is of particular interest because parallel work has demonstrated that it has excellent robustness properties regarding various network imperfections, including delay, link loss, and network fragmentation. This spatial averaging procedure has also been verified in a truly asynchronous environment, and a version exists for asynchronous lossy broadcast settings.

Our performance analysis shows a simple bound for the error transfer function which incorporates the network topology, connection density, and communication bandwidth (or messaging rate). We have shown simulations demonstrating this dependence, alongside the bounds for this error transfer function. The analytical bounds, though derived for steady-state estimation, seem to provide some useful intuition in understanding the quality of estimation in the non-stationary

[3]Obviously, 10 and 20 messages per unit time are unrealistic. We merely include these cases to show the exponential improvement of the algorithm's performance as a function of messaging rate.

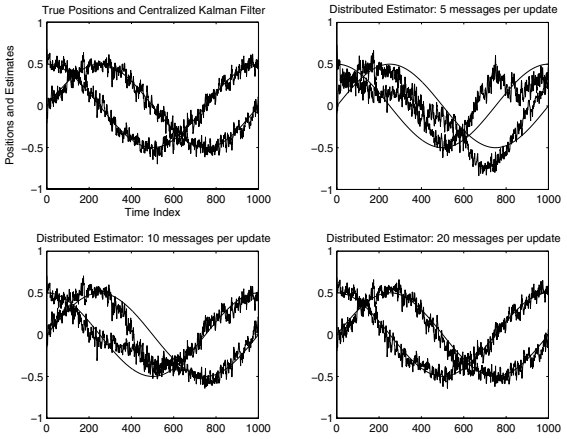

Fig. 6. The simulation results from the topology shown in Figure 5. The trajectory shown is the worst among all sensors, in the mean-square error sense. As the number of messages per unit time increases, the performance of the distributed estimator improves dramatically.

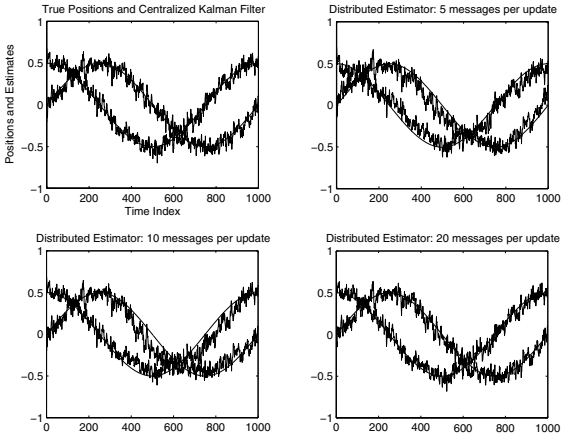

Fig. 7. The simulation results from the topology shown in Figure 5 with all "diagonal" connections added. Note that the improved communication topology has significantly improved the performance of the distributed estimation scheme.

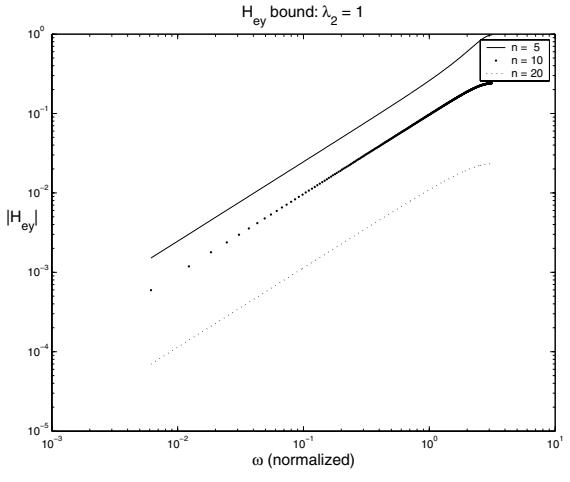

Fig. 8. The bound on the error transfer function for the network depicted in Figure 5, in logarithmic scale. Note the drastic improvement in tracking low-frequency signals as n increases.

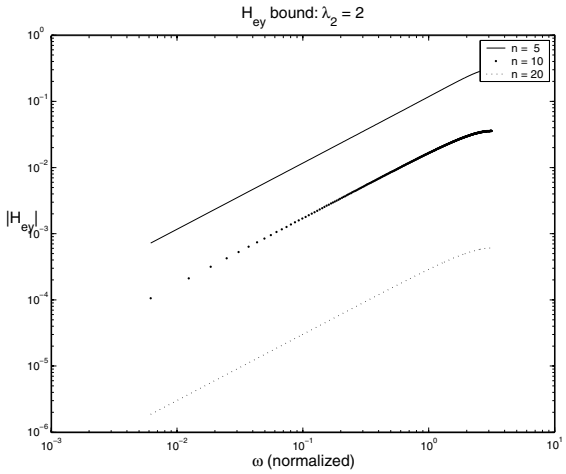

Fig. 9. The bound on the error transfer function for the network with "diagonal" connections. We see the same qualitative dependence on the number of message exchanges, but this network has significantly larger λ_2. This is verified by improved performance in the simulation results, as shown in Figure 7.

case. Further, the interpretation of the distributed estimator as a concatenation of an optimal filter with a low-pass prefilter provides additional intuition for the performance of distributed estimation as a function of the speed of the process dynamics. This is, we believe, a novel contribution to distributed estimation, which typically does not allow systematic analysis of performance under realistic network conditions. This kind of analysis and intuition may prove useful in the future for designers of large-scale sensor networks to be used for dynamic tracking applications.

The simulation results do not speak to the full power of this approach, in that this distributed filtering mechanism is naturally and simply scalable to arbitrarily large networks, while maintaining analytical performance bounds that are *directly related to the underlying sensor network*. Of course, bounds on the error transfer function are not necessarily appropriate for all applications of distributed estimation, but it is likely that this will be useful for distributed tracking applications where quantitative performance measures are essential.

REFERENCES

[1] D.P. Spanos, R. Olfati-Saber, and R. M. Murray, "Distributed sensor fusion using dynamic consensus," *(To Appear) Proceedings of IFAC World Congress*, 2005.

[2] B.S. Rao and H.F. Durrant-Whyte, "Fully decentralised algorithm for multisensor kalman filtering," *IEEE Proceedings-D*, 1991.

[3] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar, "Next century challenges: scalable coordination in sensor networks," *Proc. of Mob. Comp. and Net.*, 1999.

[4] I. Akyildiz, W. Su, Y. Sankarasubramniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, pp. 102–114, August 2002.

[5] F. Zhao, Jaewon Shin, and James Reich, "Information-driven dynamic sensor fusion," *IEEE Signal Processing*, 2002.

[6] W. R. Heinzelman, J. Kulik, and H. Balakrishnan, "Adaptive protocols for information dissemination in wireless sensor networks," *Proc. of Mob. Comp. and Net.*, 1999.

[7] R. Olfati-Saber and R. M. Murray, "Consensus problems in networks of agents with switching topology and time-delays," *IEEE Transactions on Automatic Control*, 2004.

[8] A. Jadbabaie, J. Lin, and A. S. Morse, "Coordination of groups of mobile autonomous agents using nearest neighbor rules.," *Proceedings of the Conference on Decision and Control*, 2002.

[9] M. Mehyar, D.P. Spanos, J. Pongsajapan, S.H. Low, and R.M. Murray, "Distributed averaging on a peer-to-peer network," *Submitted to CDC*, 2005.
[10] R. Merris, "Laplacian matrices of a graph: A survey," *Linear Algebra and its Applications*, 1994.
[11] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," *Proceedings of the CDC*, 2003.
[12] L. Xiao, S. Boyd, and S. Lall, "A scheme for asynchronous distributed sensor fusion based on average consensus," *(To Appear) Proceedings of IPSN*, 2005.
[13] Y. Hatano and M. Mesbahi, "Consensus over random networks," *Proceedings of the CDC*, 2004.
[14] D.P. Spanos, R. Olfati-Saber, and R.M. Murray, "Dynamic consensus on mobile networks," *(To Appear) Proceedings of IFAC World Congress*, 2005.
[15] R.Olfati-Saber, "Ultrafast consensus on small-world networks," *(To Appear) Proceedings of ACC*, 2005.